# Average-Case Bounds for the Complexity of Path-Search

Nicholas Pippenger*
(nicholas@cs.ubc.ca)

Department of Computer Science
The University of British Columbia
Vancouver, British Columbia V6T 1Z4
CANADA

**Abstract:** A channel graph is the union of all paths between a given input and a given output in an interconnection network. At any moment in time, each vertex in such a graph is either idle or busy. The search problem we consider is to find a path (from the given input to the given output) consisting entirely of idle vertices, or to find a cut (separating the given input from the given output) consisting entirely of busy vertices. We shall also allow the search to fail to find either a path or a cut with some probability bounded by a parameter called the failure probability. This is to be accomplished by sequentially probing the idle-or-busy status of vertices, where the vertex chosen for each probe may depend on the outcome of previous probes. Thus a search algorithm may be modelled as a decision tree. For average-case analysis, we assume that each vertex is independently idle with some fixed probability, called the vacancy probability (and therefore busy with the complementary probability).

For one commonly studied type channel graph, the parallel graph, we show that the expected number of probes is at most proportional to the length of a path, irrespective of the vacancy probability, and even if the allowed failure probability is zero. Another type of channel graph we study is the spider-web graph, which is superior to the parallel graph as regard linking probability (the probability that an idle path, rather than a busy cut, exists). For this graph we give an algorithm for which, as the vacancy probability is varied while the positive failure probability is held fixed, the expected number of probes reaches its maximum near the critical vacancy probability (where the linking probability make a rapid transition from a very small value to a substantial value). This maximum expected number of probes is about the cube-root of the diversity (the number of paths between the input and output).

* This research was supported by an NSERC Operating Grant.

## 1. Introduction

A *channel graph* is an acyclic directed graph $G = (V, E)$ with *vertices* $V$ and *edges* $E$ in which there exists a *source* vertex $s \in V$ and a *target* vertex $t \in V$ such that every vertex lies on a directed path from $s$ to $t$. (Such a source and target, if they exist, are clearly unique.) The vertices other than the source and target will be called *links*.

We shall deal in this paper with two particular families of channel graphs that have been proposed and analyzed for use in telephone-switching networks. The first family, comprising the *fully parallel* channel graphs $F_k$ (or simply *parallel graphs*), can be described as follows. Let $T_k$ be a tree whose vertices are the $2^{k+1} - 1$ words of length at most $k$ over the alphabet $\{0, 1\}$, with the empty word $\varepsilon$ as the root, with edges directed from each word $w$ of length at most $k - 1$ to each of the two words $w0$ and $w1$, and with the $2^k$ words of length $k$ as leaves. Let $T'_k$ be a tree like $T_k$, but with all vertices "primed" (to distinguish them from the corresponding vertices in $T_k$) and with all edges reversed. Then $F_k$ is obtained from $T_k$ and $T'_k$ by joining each leaf $w$ of $T_k$ with the corresponding leaf $w'$ of $T'_k$ by an edge $(w, w')$. The source of $F_k$ is the root of $T_k$, and the target is the root of $T'_k$. The second family, comprising the *spider-web* channel graphs $G_k$ (or simply *spider-web graphs*), is formed like $F_k$, except that each leaf $w$ of $T_k$ is joined to the leaf $\mathrm{Rev}(w)'$ of $T'_k$ by an edge $\big(w, \mathrm{Rev}(w)'\big)$, where $\mathrm{Rev}(w) = \sigma_k \cdots \sigma_1$ denotes the *reversal* of the word $w = \sigma_1 \cdots \sigma_k$.

A *state* of a channel graph is an assignment of a *status* (*busy* or *idle*) to each link of the graph. We shall extend such an assignment to all vertices by agreeing that the source and target are always idle. We shall deal in this paper with a particular probability distribution on the states of a channel graph, which again has been proposed and analyzed in the context of telephone-switching networks. We choose a real number $q$, in the range $0 < q < 1$, which we call the *vacancy probability*; its complement $p = 1 - q$ is called the *occupancy probability*. We then define a random state of a channel graph to be one in which each link is independently idle with probability $q$ or busy with probability $p$. This probability distribution on states, which was introduced independently by Lee [L1] and Le Gall [L2], is one most commonly used in the probabilistic analysis of interconnection networks.

We shall say that a channel graph is *linked* in a given state if there exists a directed path from the source to the target consisting entirely of idle links. We shall say that a channel graph is *blocked* in a given state if there exists a cut between the source and the target consisting entirely of busy links. (Clearly a channel graph in a given state is either linked or blocked, but not both.) If a channel graph $H$ is in a random state with vacancy probability $q$, the *linking probability* will be denoted $Q(H, q)$, and the complementary *blocking probability* will be denoted $P(H, q) = 1 - Q(H, q)$. For the channel graphs we consider, the asymptotic dependence of the linking probability on $q$ can be described fairly easily. For $q$ strictly less that the critical probability $q_0 = 1/\sqrt{2}$, $Q(H, q)$ tends to zero as $k$ tend to infinity for either $H = F_k$ or $H = G_k$, for the simple reason that the expected number of idle paths from the source to the target tends to zero. For $q > q_0$, the linking probability tends to a limit (as $k$ tends to infinity). This limit (which is $1 - (1-q)^4/q^4$ for $G_k$ and the smaller $1 - (1-q^2)^2/q^4$ for $F_k$) is strictly positive (for $q > q_0$), but also strictly less than unity (for $q < 1$). We shall refer to the range $0 < q \le q_0$ as the "blocking regime", and the range $q_0 < q < 1$ as the "linking regime". See Ikeno [I] for analysis applicable to parallel graphs, and Pippenger [P3] for spider-web graphs.

A few words are in order here concerning the channel graphs and the probability distribution that we are using. Channel graphs arise form the study of telephone-switching networks, in which a number of inputs are connected to a number of outputs by means of a network comprising a number of switches interconnected by wires. We model the inputs, outputs and wires (which we call links) by vertices, and individual switches by edges. (There is an alternative convention used in modelling switching networks, whereby wires are modelled by edges and aggregations of switches (called crossbars) are modelled by vertices. This convention is only applicable to networks in which the switches aggregate into crossbars. This condition is satisfied by the networks considered in this paper, but we prefer the more flexible convention that we have adopted.) Both parallel graphs and spider-web graphs are among those considered by Ikeno [I]. Although spider-web graphs are not optimal (in the sense that they do not have the largest possible linking probability among graphs arising from networks with a given number of components; see Chung and Hwang [CH]), they have been shown by Pippenger [P3] to be asymptotically optimal. In the terminology of Pippenger [P3], the parallel graphs have rhyme scheme $12\cdots(k-1)k(k-1)\cdots 21$, and the spider-web graphs have rhyme scheme $12\cdots(k-1)k1\cdots(k-2)(k-1)$. The corresponding interconnection networks have also been studied for their combinatorial properties: the parallel graphs arise from a well-known network shown to be rearrangeable by Beneš [B1]; the question of whether the network corresponding to the spider-web graph is rearrangeable is the "shuffle-exchange conjecture" (see Beneš [B2]).

Consider now an algorithm that seeks to determine whether a known channel graph in an unknown state is linked or blocked. The algorithm gathers information about the state of the channel graph by sequentially probing the status of links until all the links of either an idle path or a busy cut have been probed. (The decision as to which link to probe at any step may depend upon the outcomes of all previous probes.)

Such an algorithm can be modelled as a decision tree. The elements of such a tree will be called *nodes* and *arcs* (to distinguish them from the vertices and edges of the channel graph). Each node is either a *probe* node, in which case it is labelled with the name of a link in the channel graph and has two outgoing arcs (one labelled "idle" and the other labelled "busy") leading to other nodes, or a *leaf* node, in which case is is labelled with one of the two possible outcomes ("linked" or "blocked") and has no outgoing arcs. There is a distinguished probe node, called the *root*, that has no incoming arcs; every other node has exactly one incoming arc. Execution of the algorithm begins at the root and proceeds in an obvious way, probing links and following the appropriate arcs, until it arrives at a leaf node that announces the final result. There is an obvious notion of such an algorithm being correct: every trajectory from the root to a "linked" leaf probes a set of links corresponding to a path in the channel graph and departs each of these nodes along the "idle" arc, and every trajectory from the root to a "blocked" leaf probes a set of links corresponding to a cut in the channel graph and departs each of these nodes along the "busy" arc. We shall denote by $\mathcal{E}(H, q)$ the minimum possible expected number of probes for any algorithm that correctly searches the channel graph $H$ with vacancy probability $q$. Bounds to $\mathcal{E}(H, q)$ for series-parallel channel graphs (which include the case $H = F_k$ of parallel graphs, also considered below) have been give by Lin and Pippenger [LP].

We shall also want to consider search algorithms that occasionally fail to reach a decision as to whether the graph is linked or blocked. We shall model such algorithms by decision trees in which there are leaves of a third kind, "failure" leaves. We shall say that such an algorithm is $\varepsilon$-*correct* (for a given graph $H$ and vacancy probability $q$) if the trajectory from the root reaches a failure leaf with probability at most $\varepsilon$, and if the algorithm reaches a correct leaf whenever it does not reach a failure leaf. We shall denote by $\mathcal{E}_\varepsilon(H, q)$ the minimum possible expected number of probes for any algorithm that $\varepsilon$-correctly searches the channel graph $H$ with vacancy probability $q$. We have of course

$$\mathcal{E}_\varepsilon(H, q) \le \mathcal{E}_0(H, q) = \mathcal{E}(H, q).$$

We shall only need to allow $\varepsilon > 0$ in one situation: that of spider-web graph with $q > 1/\sqrt{2}$, where the linking probability is bounded away from zero. But in this situation the blocking probability is also bounded away from zero, and it seems difficult to object to an algorithm that fails with at most a fixed proability that is small compared with the blocking probability.

Two final comments are in order. First, we have not considered randomized search algorithms: the probability space is that of the states of the channel graph. This entails no loss of generality: since the search process is a "game against nature" (rather than against an adversary), the payoff of a mixed strategy can always be matched or exceeded by that of a pure strategy. Second, we have taken the position that $q$ is known and fixed while $k$ tends to infinity (so that the graphs $F_k$ and $G_k$ become larger and larger). This of course does not correspond to the situation in practice where a specific network is subjected to varying traffic. It does accord well, however, with the fact that it is usually desired to operate a network of the type considered at a vacancy probability $q$ just slightly larger than the critical probability $q_0 = 1/\sqrt{2}$, which is independent of $k$; this allows the network to carry the largest amount of traffic consistent with a blocking probability bounded away from unity. (The provision of a blocking probability sufficiently close to zero to be satisfactory to subscribers is the task of additional stages of switching, outside the network considered here and beyond the scope of this paper.) Finally, we observe that in a situation in which the vacancy probability is not known, it is of course possible to estimate it closely and with high confidence by sampling.

## 2. Parallel Graphs

In this section we shall study the expected search cost $\mathcal{E}(F_k, q)$ of the fully parallel channel graph $F_k$. The following is our main result.

*Theorem 2.1:* For all $k \geq 1$ and all $0 < q < 1$, we have

$$\mathcal{E}(F_k, q) \leq 4k.$$

Observe that this bound is independent of $q$. Since there are just $2k$ ranks of links in $F_k$, on average at most 2 links in each rank need to be probed.

Let $\mathcal{E}'(F_k, q)$ denote the expected search cost of $F_k$ when the source and target are, like the links, independently busy with probability $p$. (In this definition, say that $F_k$ is blocked if either the source or target is busy.) We have the inequality

$$\mathcal{E}(F_k, q) \leq 2\mathcal{E}'(F_{k-1}, q), \tag{2.1}$$

since apart from its source and target $F_k$ comprises two disjoint copies of $F_{k-1}$. Thus Theorem 2.1 will follow from (2.1) and the following proposition.

*Proposition 2.2:* For all $k \geq 0$ and all $0 < q < 1$, we have

$$\mathcal{E}'(F_k, q) \leq 2(k+1).$$

In the case $k = 0$, $F_0$ is a single edge, joining the source to the target. Thus we have

$$\mathcal{E}'(F_0, q) \leq 2. \tag{2.2}$$

We also have the recurrence

$$\mathcal{E}'(F_k, q) \leq 2 + q^2\big(1 + P(F_{k-1}, q)\big)\mathcal{E}'(F_{k-1}, q), \tag{2.3}$$

since we can search $F_k$ by the following algorithm. First probe the source and target, at cost 2. Halt with the result "blocked" unless both the source and target are idle, an event which occurs with probability $q^2$. In this case, search one of the two disjoint copies of $F_{k-1}$, at expected cost $\mathcal{E}'(F_{k-1}, q)$. Halt with the result "linked" unless the searched copy is blocked, an event which occurs with probability $P(F_{k-1}, q)$. In this case, search the remaining copy of $F_{k-1}$. This algorithm establishes (2.3).

To use this recurrence, we need an estimate for the blocking probability $P(F_k, q)$, which is provided by the following lemma.

*Lemma 2.3:* For all $k \geq 0$ and all $0 < q < 1$, we have

$$P(F_k, q) \leq q^{-2} - 1.$$

*Proof:* We proceed by induction on $k$. For the basis, $k = 0$, we have

$$\begin{aligned} P(F_0, q) &= 1 - q^2 \\ &\leq q^{-2} - 1, \end{aligned}$$

where the inequality, which is equivalent to $1 \leq (q^2 + q^{-2})/2$, compares the geometric and arithmetic means of $q^2$ and $q^{-2}$.

For the inductive step, $k \geq 1$, we have

$$\begin{aligned} P(F_k, q) &= 1 - q^2\big(1 - P(F_{k-1}, q)^2\big) \\ &= 1 - q^2 + q^2 P(F_{k-1}, q)^2 \\ &\leq 1 - q^2 + q^2\big(q^{-2} - 1\big)^2 \\ &= 1 - q^2 + q^2\big(q^{-4} - 2q^{-2} + 1\big) \\ &= q^{-2} - 1, \end{aligned}$$

where the inequality follows by inductive hypothesis. $\triangle$

Substituting this estimate into (2.3) yields the recurrence

$$\mathcal{E}'(F_k, q) \leq 2 + \mathcal{E}'(F_{k-1}, q). \tag{2.4}$$

Proposition 2.2 now follows by induction, with (2.2) as the basis and (2.4) as the inductive step.

The foregoing algorithm and its analysis have a simple interpretation that will be useful later. Consider the problem of finding an idle path from the root to an idle leaf of $T_k$, supposing as usual that the root is idle and that every other vertex is independently idle with probability $q$. This problem can be solved by an obvious "depth-first search", and the analysis given above yields the following result.

*Corollary 2.4:* If the root of $T_k$ is idle, and every other vertex is independently idle with probability $q$, then depth-first search finds an idle path from the root to an idle leaf (or finds a busy cut showing that no such path exists) using at most $2k$ probes (and finds a cut rather than a path with probability at most $(1-q)^2/q^2$).

If we fold the graph $F_k$ back upon itself, we obtain a tree similar to $T_k$, which we shall denote $T_k''$, with two vertices of $F_k$ corresponding to each vertex of $T_k$. Thus Theorem 2.1 can be recovered from Corollary 2.4 by observing that each probe in $T_k''$ can be simulated by 2 probes in $F_k$ (so that we have a bound of $4k$ rather than $2k$), and 2 vertices in $F_k$ must be idle for the corresponding vertex in $T_k''$ to be idle (so that in the bound $(1-q)^2/q^2$, $q$ must be replaced by $q^2$ to obtain $(1-q^2)^2/q^4$).

## 3. Spider-Web Graphs

In this section we shall begin our study of the expected search cost $\mathcal{E}(G_k, q)$ of the spider-web channel graph $G_k$. We shall present a algorithm that performs well in the blocking regime, and in the next section we shall combine this algorithm with other elements to obtain good performance in the linking regime as well.

We can search $G_k$ using an algorithm comprising the following sequence of $k$ phases. During the $j$-th phase, for $1 \leq j \leq k$, we shall probe each link $w$ in ranks $j$ and $j'$ that meets the following condition: there exists a path from the source through $w$ to the target on which every link that has been probed is idle.

This algorithm will always find either an idle path or a busy cut. If there is an idle path from the source to the target, all its links will be probed. (Thus this algorithm will probe the links on all idle paths, and not just one.) If a path from the source to the target has a busy link, then such a link will be probed in the $j$-th phase, where $j$ is the smallest number such that the path has a busy link in rank $j$ or $j'$. Thus if every path has a busy link, the algorithm will probe every link of some busy cut.

It remains to analyze the expected cost of this algorithm, which we shall denote $A(k,q)$. We may write

$$A(k,q) = \sum_{1 \le j \le k} A_j(k,q),$$

where $A_j(k,q)$ denotes the expected number of probes in the $j$-th phase. The expected number of probes in the $j$-th phase may be expressed as the sum over links $w$ in rank $j$ or $j'$ of the probability that $w$ is probed. There are $2^j$ links in rank $j$ and $2^j$ links in rank $j'$, and thus there are $2 \cdot 2^j$ links i rank $j$ or $j'$. If $v$ and $w$ are two links in rank $j$, then the probability that $v$ is probed is the same as the probability that $w$ is probed. To see this, observe that there is an automorphism of $G_k$ taking the union of all paths from the source through $v$ to the target to the union of all paths from the source through $w$ to the target, and that the probability distribution on states is invariant under automorphisms of $G_k$. Furthermore, the probability that a link $w$ is probed is the same as the probability that $w'$ is probed. This can be seen by considering anti-automorphisms, which exchange source and target and reverse the directions of edges. Denoting this common probability of being probed by $P_j(k,q)$, we have

$$A(k,q) = 2 \sum_{1 \le j \le k} 2^j P_j(k,q).$$

To estimate $P_j(k,q)$, consider a link $w$ in rank $j$. There are $j-1$ links on the path in $T_k$ from the source to $w$ (not including $w$), and all of these links must be idle if $w$ is to be probed; this event occurs with probability $q^{j-1}$. Thus $P_j(k,q) \le q^{j-1}$, and we have

$$A(k,q) \le 2 \sum_{1 \le j \le k} 2^j q^{j-1}.$$

There are $k$ terms in this summation; if $q \le 1/2$, the largest of these is 2, which arises from $j = 1$. Thus we have (for $q \le 1/2$)

$$A(k,q) \le 4k. \tag{3.1}$$

To analyze this algorithm for $1/2 < q < 1$, we shall need a sharper estimate for $P_j(k,q)$. Consider a link $w$ in rank $j$. There are $2^{k-j}$ paths from the source through $w$ to the target. Denote by $K$ the union of these paths. The portion of $K$ between the source and $w$ is a path containing $j-1$ links (not including $w$). These $j-1$ links have already been probed (in phases earlier than the $j$-th), and if $w$ is to be probed they must all be idle; this event occurs with probability $q^{j-1}$. Now consider the portion of $K$ joining $w$ to the target. The links of this portion that have already been probed are those in ranks $(j-1)', \ldots, 2', 1'$. They form a tree (a subtree of $T_k'$) that is the union of at most $2^{k-j}$ paths each containing $j-1$ links, and if $w$ is to be probed all the links on at least one such path must be idle; this event occurs with probability at most $2^{k-j}q^{j-1}$ (and of course with probability at most 1). Thus we have

$$P_j(k,q) \leq q^{j-1} \min\{1, 2^{k-j}q^{j-1}\},$$

and therefore

$$A(k,q) \leq 2 \sum_{1 \leq j \leq k} 2^j q^{j-1} \min\{1, 2^{k-j}q^{j-1}\}.$$

Since we now assuming $q > 1/2$, we can write

$$A(k,q) \leq 8 \sum_{1 \leq j \leq k} 2^j q^j \min\{1, 2^{k-j}q^j\}.$$

To estimate this sum, we define $\alpha(q)$ by

$$\alpha(q) = \frac{\log 2}{\log(2/q)},$$

so that $1 < 2^{k-j}q^j$ if and only if $j < \alpha(q)k$. Thus we have

$$A(k,q) \leq 8 \sum_{1 \leq j < \alpha(q)k} 2^j q^j + 8 \sum_{\alpha(q)k \leq j \leq k} 2^k q^{2j}.$$

The two summations contain $k$ terms between them, and each of these terms is at most the maximum of these terms. In the case of the first summation, this maximum is (since $2q > 1$) at most $(2q)^{\alpha(q)k} = 2^{\beta(q)k}$, where we define

$$\beta(q) = \frac{\log(2q)}{\log(2/q)}$$

for $q > 1/2$, and in the case of the second summation it is (since $q^2 < 1$) at most $2^k q^{2\alpha(q)k} = 2^{\beta(q)k}$. Thus we have (for $1/2 < q < 1$)

$$A(k,q) \leq 8k\, 2^{\beta(q)k}. \tag{3.2}$$

We combine the results of (3.1) and (3.2) in the following theorem.

*Theorem 3.1:* For all $k \geq 1$ and $0 < q < 1$, we have

$$\mathcal{E}(G_k, q) \leq 8k\, 2^{\beta(q)k},$$

where

$$\beta(q) = \begin{cases} 0, & \text{for } 0 < q \leq 1/2; \\ \frac{\log(2q)}{\log(2/q)}, & \text{for } 1/2 < q < 1. \end{cases}$$

This is achieved by an algorithm that finds *all* idle paths from the source to the target.

We shall now restate the results of this section in a form that simplifies the expressions involved and gives them some intuitive significance. We may regard the search for an idle path as taking place in a search-space containing $2^k$ paths. We shall think of the subset of paths actually probed as constituting a subspace of fractional dimension: if $2^{dk}$ paths are probed, then the dimension of this subspace is $d$ $(0 \leq d \leq 1)$. This suggests that we transfer attention from $\mathcal{E}(G_k, q)$ to $k^{-1} \log \mathcal{E}(G_k, q)$. Furthermore, it will be convenience to make the substitution $q = 2^\lambda$ as well, so that $-\infty < \lambda < 0$. Thus we define

$$d(k, \lambda) = k^{-1} \log \mathcal{E}(G_k, 2^\lambda).$$

We can now express the analysis in this section by

$$d(k, \lambda) \leq b(\lambda) + O\left(\frac{\log k}{k}\right),$$

where we define $b(\lambda)$ by

$$b(\lambda) = \begin{cases} 0, & \text{for } -\infty < \lambda \leq -1; \\ \frac{1+\lambda}{1-\lambda}, & \text{for } -1 < \lambda < 0. \end{cases}$$

We observe that $(1+\lambda)/(1-\lambda)$ increases from 0 to 1 as $\lambda$ increases from $-1$ to 0, and that it reaches the value $1/3$ at $\lambda = -1/2$. In the next section we shall derive for $-1/2 < \lambda < 0$ a sharper upper bound to $d(k, \lambda)$ (more precisely, to an analogous quantity $d_\varepsilon(k, \lambda)$ that allows for positive, but arbitrarily small, probability $\varepsilon$ of failure) that decreases from $1/3$ to 0 as $\lambda$ increases from $-1/2$ to 0. Thus the value $1/3$ represents the maximum "search dimension" required by our algorithms, achieved for the value $\lambda = -1/2$ corresponding to the critical vacancy probability $q = q_0 = 1/\sqrt{2}$.

## 4. Spider-Web Graphs in the Linking Regime

In this section we shall present an algorithm that improves the performance in the linking regime of the algorithm presented in the preceding section. The latter algorithm is limited by the fact that it finds all idle paths, so that it cannot use fewer probes than the number of such paths. We might contemplate reducing the number of idle paths explored by constraining the initial and final segment of the path. If the length of these constrained segments is chosen appropriately, the number of idle path will be reduced to a small number, just large enough to give a blocking probability near $P(G_k, q)$. The problem with this strategy is that the task of finding suitable initial and final segments also contributes a blocking probability near $P(G_k, q)$. The solution to this problem is to allow a number of alternatives (about $k$) for the initial and final segments. We shall present the resulting algorithm as a sequence of three steps, interspersed with their analysis.

Set

$$i = \lceil 4 \log(2k) \rceil.$$

*Step 1:* Find the set $I$ of all links $v$ in rank $i$ of $T_k$ such that all links on the path from the source to $v$ (including $v$ itself) are idle. Similarly, find the set $I'$ of all links $v'$ in rank $i'$ of $T'_k$ such that all links on the path from $v'$ to the target (including $v'$ itself) are idle. If either $I$ or $I'$ is empty, there is a busy cut separating the source from the target. If neither $I$ not $I'$ is empty, but either $I$ or $I'$ contains fewer than $2k$ links, the algorithm fails. Otherwise, let $v_1, \ldots, v_{2k}$ be $2k$ distinct links in $I$, and let $v'_1, \ldots, v'_{2k}$ be $2k$ distinct links in $I'$.

The number of links probed in Step 1 is at most

$$2 + 2^2 + \cdots + 2^i \leq 64k^4.$$

To estimate the probability of failure in Step 1, we need to estimate the probability that $I$ contains at least 1 but fewer than $2k$ links. The cardinality of $I$ is the population $Z_i$ in the $i$-th generation of a branching process that has been analyzed by Pippenger [P3]. By the analysis preceding Lemma 8.1 of that paper, we have

$$\Pr(1 \leq Z_i < 2k) = O\big((2k/R)^\alpha\big),$$

where $R = (2q)^i \geq (2q_0)^i \geq 4k^2$, and $\alpha = -\log\big(2(1-q)\big)/\log(2q) \geq -\log\big(2(1-q_0)\big)/\log(2q_0) > 1$. Thus we have

$$\Pr(1 \leq Z_i < 2k) = O(1/k).$$

The probability of failure in Step 1 is also $O(1/k)$, since it is at most $\Pr(1 \le Z_i < 2k)$ plus an equal contribution from the primed side of the graph.

Set
$$j = \left\lfloor \frac{k \log(2q^2) - 4 \log k}{2 \log(2q)} \right\rfloor .$$
We shall assume that $k$ is sufficiently large that $j > i$, which we may do since $\log(2q^2)/\log(2q) > 0$ for $q > q_0$. Furthermore, we shall assume that $j < (k-1)/2$, which we may do since $\log(2q^2)/\log(2q) < 1/2$ for $q < 1$.

*Step 2:* For each link $v_h$ $(1 \le h \le 2k)$ found in Step 1, find if possible using depth-first search a link $w$ in rank $j$ of $T_k$ such that all links on the path from $v_h$ to $w$ (including $w$ itself) are idle. If fewer than $k$ of these $2k$ attempts are successful, the algorithm fails. Otherwise, let $w_1, \ldots, w_k$ be $k$ such links, which then have the property that all the links on the path from the source to $w_h$, inclusive, are idle. Similarly, for each link $v'_h$ $(1 \le h \le 2k)$ found in Step 1, find if possible using depth-first search a link $w'$ in rank $j'$ of $T'_k$ such that all links on the path from $w'$ (including $w'$ itself) to $v'_h$ are idle. If fewer than $k$ of these $2k$ attempts are successful, the algorithm fails. Otherwise, let $w'_1, \ldots, w'_k$ be $k$ such links, which then have the property that all the links on the path from $w'_h$, inclusive, to the target are idle.

According to Corollary 2.4, the expected number of probes required for each of the $2k + 2k = 4k$ trials in Step 2 is at most $2(j - i) \le 2k$. Thus the expected number of links probed in Step 2 is at most $8k^2$. To estimate the probability of failure in Step 2, we must estimate the probability of having fewer than $k$ successes in $2k$ independent trials. According to Corollary 2.4, each trial succeeds with probability at least $1 - (1-q)^2/q^2 \ge 1 - (1-q_0)^2/q_0^2 > 3/4$. By Chebyshev's inequality, the probability $\Pr(Y < k)$ that the number $Y$ of successes in $2k$ trials, which has expectation at least $(3/4)\,2k = 3k/2$ and variance at most $(1/4)\,2k = k/2$, deviates by at least $3k/2 - k = k/2$ from its expectation is at most $(k/2) \;/\; (k/2)^2 = 2/k$. The probability of failure in Step 2 is thus at most $4/k$, since it is $\Pr(Y < k)$ plus a equal contribution from the primed side of the graph.

For $1 \le h \le k$, denote by $L_h$ the channel graph comprising all vertices of $G_k$ lying on paths from $w_h$ to $w'_h$, with $w_h$ as source and $w'_h$ as target. The graph $L_h$ is a generalized spider-web graph of a type dealt with by Pippenger [P3]. Since $G_k$ arises from a network with rhyme scheme $12\cdots(k-1)k12\cdots(k-2)(k-1)$, $L_h$ arises from an network with rhyme scheme $(j+1)(j+2)\cdots(k-1)k12\cdots(k-2-j)(k-1-j)$. This is equivalent to the rhyme scheme $12\cdots(s+t-1)(s+t)12\cdots(s+2t-1)(s+2t)$, where $s = 2j+1$ and $t = k-1-2j$. The channel graph arising from a network with this rhyme scheme will be denoted $G_t \circ H_s$; it is obtained from $G_t$ by substituting for each central edge $\bigl(u, \mathrm{Rev}(u)'\bigr)$ a copy of a channel graph $H_s$ that has a unique path from the source $u$ through $s$ edges and $s-1$ intermediate links to the target $\mathrm{Rev}(u)'$. Thus each $L_h$ is isomorphic to $G_t \circ H_s$. Furthermore, for $1 \le f < g \le k$, the graphs $L_f$ and $L_g$ are disjoint: the portions in $T_k$ are disjoint because $w_f$ and $w_g$ are distinct, and the portions in $T'_k$ are disjoint because $w'_f$ and $w'_g$ are distinct.

*Step 3:* For $1 \le h \le k$, determine in the following way if there is an idle path in $L_h$ from $w_h$ to $w'_h$. To determine if $L_h \approx G_t \circ H_s$ is linked, apply the algorithm of Section 3 to $G_t$ to find the set $U$ of central edges of idle paths in $G_t$. Then, for each edge $\bigl(u, \mathrm{Rev}(u)'\bigr)$ in $U$, determine if all links in the copy of $H_s$ from $u$ to $\mathrm{Rev}(u)'$ are idle, so that the idle path through $\bigl(u, \mathrm{Rev}(u)'\bigr)$ in $G_t$ extends to an idle path in $L_h$. If an idle path through some $L_h$ is found, it extends to an idle path through $G_k$. If all $L_h$ are blocked, the algorithm fails.

According to Theorem 3.1, each application of the algorithm of Section 3 to a copy of $G_t$ uses and expected number of probes at most $8t\, 2^{\beta(q)t} \le 8k\, 2^{\beta(q)(k-2j)}$. This expression of course also bounds the number of idle paths found, and thus the expected cardinality of $U$. For each edge in $U$, the number of probes needed to search the corresponding copy of $H_s$ is at most $s-1 \le k-1$. Thus the expected number of probes needed to search each $L_h$ is at most $8k^2\, 2^{\beta(q)(k-2j)}$. Since there are $k$ such channel graphs $L_h$, the expected number of links probed in Step 3 is at most $8k^3\, 2^{\beta(q)(k-2j)}$.

From the definition of $j$ we have

$$\begin{aligned} k - 2j &\le k - \frac{k\log(2q^2) - 4\log k}{\log(2q)} + 2 \\ &= \frac{k\log(1/q) + 4\log k + 2\log(2q)}{\log(2q)}. \end{aligned}$$

Since $\beta(q) = \log(2q) \,/\, \log(2/q)$ for $q > q_0$, this yields

$$\beta(q)\,(k-2j) \le \frac{k\log(1/q) + 4\log k + 2\log(2q)}{\log(2/q)},$$

and since $\log(2q) \,/\, \log(2/q) < 1 \,/\, \log(2/q) < 1$ for $q_0 < q < 1$, we obtain

$$\beta(q)\,(k-2j) \le \gamma(q)\,k + 2 + 4\log k,$$

where we define $\gamma(q) = \log(1/q) \,/\, \log(2/q)$ for $q_0 < q < 1$. Thus the expected number of links probed in Step 3 is at most

$$8k^3\, 2^{\beta(q)(k-2j)} \le 32k^7\, 2^{\gamma(q)k}.$$

To estimate the probability of failure in Step 3, we need to estimate the probability of blocking in each $L_h$. To do this we shall need to estimate the expected number $J = 2^t q^{s+2t-1} = 2^{k-2j-1} q^{2k-2j-2} \geq 2^{k-2j} q^{2k-2j}$ of idle paths from $w_h$ to $w'_h$ in $L_h$. Since

$$2j \leq \frac{k \log(2q^2) - 4 \log k}{\log(2q)},$$

we obtain $J \geq k^4$. According to Theorem 10.1 from Pippenger [P3], the blocking probability of each $L_h$ is then $1-(1-\xi)^2 + O(J^{-\beta})$, where $\xi = (1-q)^2/q^2$ and $\beta = \alpha/2(1+\alpha) > 1/4$ (since $\alpha > 1$). Since $\xi = (1-q)^2/q^2 \leq (1-q_0)^2/q_0^2 < 1/4$, we have $1-(1-\xi)^2 < 1/2$. Thus the blocking probability in each $L_h$ is at most $1/2 + O(1/k)$, and we may assume that $k$ is sufficiently large that it is at most $2/3$. Since failure occurs in Step 3 only if blocking independently occurs in all $k$ disjoint channel graphs $L_h$, the probability of failure in Step 3 is at most $(2/3)^k$.

Combining the expected number of probes from the three steps yields

$$64k^4 + 8k^2 + 32k^7\, 2^{\gamma(q)k} \leq 128k^7\, 2^{\gamma(q)k}.$$

Combining the estimates for probability of failure yields

$$O(1/k) + 4/k + (2/3)^k = O(1/k),$$

and thus for any fixed $\varepsilon > 0$ we may assume that $k$ is large enough that the probability of failure for the entire algorithm is at most $\varepsilon$. Thus we have obtained the following result.

*Theorem 4.1:* For any fixed $q > q_0$ and $\varepsilon > 0$, and for all sufficiently large $k$, we have

$$\mathcal{E}_\varepsilon(G_k, q) \leq 128k^7\, 2^{\gamma(q)k}.$$

Combining this result with Theorem 3.1, and extending the domain of definition of $\gamma(q)$, we have the following result.

*Theorem 4.2:* For any fixed $0 < q < 1$ and $\varepsilon > 0$, and for all sufficiently large $k$, we have

$$\mathcal{E}_\varepsilon(G_k, q) \leq 128k^7\, 2^{\gamma(q)k},$$

where

$$\gamma(q) = \begin{cases} 0, & \text{for } 0 < q \leq 1/2; \\ \frac{\log(2q)}{\log(2/q)}, & \text{for } 1/2 < q < q_0; \\ \frac{\log(1/q)}{\log(2/q)}, & \text{for } q_0 < q < 1. \end{cases}$$

Finally, if we define

$$d_\varepsilon(k,\lambda) = k^{-1} \log \mathcal{E}_\varepsilon(G_k, 2^\lambda)$$

as the search dimension analogous to $d(k,\lambda)$, then can express the analysis in this section by

$$d_\varepsilon(k,\lambda) \le c(\lambda) + O\left(\frac{\log k}{k}\right),$$

where we define $c(\lambda)$ by

$$c(\lambda) = \begin{cases} 0, & \text{for } -\infty < \lambda \le -1; \\ \frac{1+\lambda}{1-\lambda}, & \text{for } -1 < \lambda < -1/2; \\ \frac{-\lambda}{1-\lambda}, & \text{for } -1/2 < \lambda < 0. \end{cases}$$

As described before, the function $c(\lambda)$ assumes its maximum value $1/3$ for the unique critical value $\lambda = -1/2$.

## 5. Conclusion

The most obvious problem left by this paper is to determine whether the algorithms presented for spider-web networks are asymptotically optimal. We can show that this is the case in the blocking regime, but in the linking regime the best lower bound we have been able to obtain falls below the upper bound. It is also an open question as to whether the upper bound we have presented in the linking regime can be obtained for $\varepsilon = 0$.

It would also be of interest to address some of the shortcomings of the models we have employed. The probability distribution on states introduced by Lee [L1] and Le Gall [L2] is the simplest to use for analysis, but is somewhat unrealistic. Some of its defects are rectified in a more complicated model introduced by Pippenger [P1] (and in a slightly different form by Koverninskiĭ [K]). This model has been used to analyze the linking probabilities of both series-parallel networks (Pippenger [P1]) and spider-web networks (Pippenger [P2]). The decision-tree model we have used gives a good account of time requirements (at least in so far as they are reflected by probes), but takes no account of space requirements. This shortcoming could be addressed by replacing the decision-tree with a decision-graph, and reckoning space as the logarithm of the number of nodes in the graph. It should then be possible to analyze the time-space trade-off for searching spider-web networks.

Finally, we should mention the possibility of extending the analysis to other families of channel graphs. The spider-web graphs are superior to parallel graphs as regards linking probability, but they require more time for path-search. This raises the problem of constructing a new family of channel graphs that might enjoy the advantages of both, and the disadvantages of neither. Since the linking probability is determined largely by events in the outermost stages of a network, while much of the search time is consumed by links deeper in the network, it seems reasonable to hope that a network that is spider-web-like in its outer stages and parallel-like within (say with the rhyme scheme $12\cdots j(j+1)\cdots(k-1)k(k-1)\cdots(j+1)1\cdots(j-1)j$, for some $j$ growing very slowly with $k$) would accomplish these objectives.

### 6. References


[B1] V. E. Beneš, "Optimal Rearrangeable Multistage Connecting Networks", *Bell System Tech. J.*, 43 (1964) 1641–1656.

[B2] V. E. Beneš, "Proving the Rearrangeability of Connecting Networks by Group Calculations", *Bell System Tech. J.*, 45 (1975) 421–434.

[CH] F. R. K. Chung and F. K. Hwang, "The Connection Patterns of Two Complete Binary Trees", *SIAM J. Algeb. and Discr. Meth.*, 1 (1980) 322–335.

[I] N. Ikeno, "A Limit on Crosspoint Number", *IRE Trans. on Inform. Theory*, 5 (1959) 187–196.

[K] I. V. Koverninskiĭ, "Estimation of the Blocking Probability for Switching Circuits by Means of Probability Graphs", *Problems of Info. Transm.*, 11 (1975) 63–71.

[L1] C. Y. Lee, "Analysis of Switching Networks", *Bell Syst. Tech. J.*, 34 (1955) 1287–1315.

[L2] P. Le Gall, "Étude du blocage dans les systèmes de commutation téléphoniques automatiques utilisant des commutateurs électroniques du type crossbar", *Ann. des Téécomm.*, 11 (1956) 159–171, 180–194.

[LP] G. Lin and N. Pippenger, "Routing Algorithms for Switching Networks with Probabilistic Traffic", *Networks*, 28 (1996) 21–29.

[P1] N. Pippenger, "On Crossbar Switching Networks", *IEEE Trans. on Comm.*, 23 (1975) 646–659.

[P2] N. Pippenger, "The Blocking Probability of Spider-Web Networks", *Random Structures and Algorithms*, 2 (1991) 121–149.

[P3] N. Pippenger, "The Asymptotic Optimality of Spider-Web Networks", *Discr. Appl. Math.*, 37/38 (1992) 437–450.